LA-UR-85-413

CONF-850266--2

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36

LA-UR--85-413

DE85 007699

TITLE: FUNDAMENTALS OF PLASMA SIMULATION

AUTHOR(S): David W. Forslund

SUBMITTED TO Second International School for Space Simulations
Honolulu, Hawaii, February 3-16, 1985

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Los Alamos Los Alamos National Laboratory
Los Alamos, New Mexico 87545

FUNDAMENTALS OF PLASMA SIMULATION

David W. Forslund
Applied Theoretical Physics
Los Alamos National Laboratory
Los Alamos, New Mexico 87545

ABSTRACT. With the increasing size and speed of modern supercomputers, the incredibly complex nonlinear properties of plasmas in the laboratory and in space are being successfully explored in increasing depth. Of particular importance have been numerical simulation techniques involving finite size particles on a discrete mesh. After discussing the importance of this means of understanding a variety of nonlinear plasma phenomena, we describe the basic elements of particle-in-cell simulation and their limitations and advantages. The differencing techniques, stability and accuracy issues, data management and optimization issues are discussed by means of a simple example of a particle-in-cell code. Recent advances in simulation methods allowing large space and time scales to be treated with minimal sacrifice in physics are reviewed. Various examples of nonlinear processes successfully studied by plasma simulation will be given.

1. Introduction

The many degrees of freedom in plasmas leads to an incredible variety of plasma phenomena on a spectrum of space and time scales. Nonlinear phenomena in plasmas are known to have many more diverse properties than even compressible fluid dynamics. Various theoretical models have been developed to describe the observed plasma properties, but these, of necessity, require many simplifying assumptions whose validity is many times questionable and difficult to verify. Although weak turbulence theory has been successful over a limited range of parameters, frequently the nonlinearity of some plasma process is so strong that it breaks down. It is in this regime that plasma simulations are the most useful and the easiest to perform. In regimes where weak turbulence theory is reasonably accurate, particle simulation is the most expensive and difficult to perform. Thus there has been considerable complementarity between weak turbulence theory and plasma simulation, each having its own domain of importance.

For more than 15 years the simulation of plasma phenomena on large scale computers has played an important role in explaining many observed plasma phenomena and in verifying or disproving various nonlinear plasma theories. The large variation in time and space scales present in typical plasma problems has continued to present a challenge to the simulationist. Today, as computers continue to increase in size and speed the nonlinear plasma problems that can be studied on the computer have become even more realistic. As an introduction to computer simulation techniques of plasmas, we will review the basic methods and illustrate their use on problems of significance.

2. Basic Equations

2.1 Liouville Equation

The equations usually used to describe a plasma are Newton's laws and Maxwell's equations:

$$
\begin{aligned}
\frac{d\vec{v}_j}{dt} &= \frac{q_j}{m_j}\left(\vec{E} + \frac{\vec{v}_j \times \vec{B}}{c}\right) \\
\frac{d\vec{x}}{dt} &= \vec{v} \\
\nabla \times \vec{E} &= -\frac{1}{c}\frac{\partial \vec{B}}{\partial t} \\
\nabla \cdot \vec{B} &= 0, \quad \nabla \cdot \vec{E} = 4\pi\rho \qquad (1) \\
\nabla \times \vec{B} &= \frac{4\pi\vec{j}}{c} + \frac{1}{c}\frac{\partial \vec{E}}{\partial t} \\
\rho(\vec{x}) &= \sum_j q_j \delta(\vec{x} - \vec{x}_j) \\
\vec{j}(\vec{x}) &= \sum_j q_j \vec{v}_j \delta(\vec{x} - \vec{x}_j)
\end{aligned}
$$

Almost all classical plasma physics involves the solving of these equations with various approximations. The simultaneous solution of these equations for a large collection of particles is equivalent to the solution of the Liouville equation for the N body distribution function. Solutions to these equations exhibit collisional effects determined by the structure of the δ functions in the above equations. Usually one speaks of a plasma as having a large number of particles per Debye sphere, in which case only terms to the lowest order in this plasma parameter are kept.

2.2 Vlasov Equations

In this collisionless limit, we have the Vlasov equation system:

$$
\frac{\partial f}{\partial t} + \vec{v} \cdot \frac{\partial f}{\partial \vec{x}} + \frac{q}{m}\left(\vec{E} + \frac{\vec{v} \times \vec{B}}{c}\right) \cdot \frac{\partial f}{\partial \vec{v}} = 0
$$

$$
\begin{aligned}
\nabla \cdot \vec{E} &= 4\pi \int q f d^3 \vec{v}, \\
\nabla \times \vec{B} &= \frac{1}{c}\frac{\partial \vec{E}}{\partial t} + \frac{4\pi}{c} \int q f \vec{v} d^3 \vec{v}, \qquad (2) \\
\nabla \cdot \vec{B} = 0, \quad \nabla \times \vec{E} &= -\frac{1}{c}\frac{\partial \vec{B}}{\partial t}.
\end{aligned}
$$

Although much successful effort has gone into solving Eqs. (2) for the distribution function directly[*Sakanaka, et al., 1971; Denavit, 1972*], the methods have been difficult or expensive to extend to 2 dimensions. The traditional form of plasma simulation [*Buneman, 1967*] involving particles is equivalent to solving the Vlasov equation by the method of characteristics, where the particle orbits are the characteristics of the Vlasov equation. Since the particles have finite size, however, they exhibit properties of the original Liouville system from which the Vlasov equation is derived. Consequently, plasma simulation codes exhibit many kinetic properties of plasmas including bremsstrahlung, collisional excitation and damping of plasma waves. With care one can study these phenomena directly with a simulation code. However, usually the collisions are not desireable, so one must take care to reduce them .

2.3 General Methods

To solve Eqs. (2) one must represent the spatial coordinate as a discrete grid and advance the equations over discrete time intervals or steps. Stability and accuracy conditions dictate the choice of the space and time steps. Early methods in one dimension treated particles as points moving through each other. However, because one cannot have as many particles per Debye sphere in the computer as one has in most plasmas, this method results in an excessively high noise level (and hence collisional level). To reduce this problem, particles are made to be of finite size and to deposit information onto and take information from a grid. In other words the δ functions of Eqs. (1) are made finite in width. This drastically decreases the noise level by significantly altering the collisional process. Particles no longer undergo large angle collisions from a close approach but experience only small angle collisions as they pass through one another. As a consequence, the finite size particle simulation method is most suited for studying collisionless plasmas. In this limit collisions can be added back in a controlled manner. To the lowest order then one can view the particle-in-cell simulation technique as a Monte Carlo solution to the Vlasov equation by integration along its characteristics. This Monte Carlo aspect of the solution illustrates why problems that are sensitive to the behavior of particles in a small region of velocity space are difficult to handle. Problems that depend primarily on the bulk plasma properties, albeit high nonlinear, are most readily solved by this technique.

The solution to Eqs. 2 are obtained by integrating in a series of discrete time steps the orbits of a large number of finite size charged particles (typically 10^4 10^6). The lagrangian positions of the particles are used to deposit the charge and currents onto the fixed discrete grid. The field equations are then solved on this discrete grid (typically 10^3 10^5 cells) to obtain the self-consistent fields for the next time step. The discrete field points are then interpolated to form a continuous field variable to advance the particle orbit. The two important numerical aspects are the space and time differencing and the interpolation of the fields from the grid to the particle and the particle charge onto the grid (i.e. the shape of the particle). A variety of techniques have been employed successfully and I will discuss only a few.

3. A One Dimensional Electrostatic Code

3.1 Basic Equations

To clarify the preceding general introduction, let us consider what is involved in writing a simple one dimensional electrostatic plasma simulation code. As an interesting generalization of the usual one dimensional code we consider the equations for a spherical plasma and include the case of a space and time varying grid on which the equations are solved. The basic equations to be solved are:

$$\frac{1}{r^2}\frac{d}{dr}r^2E_r = 4\pi\rho \tag{3}$$

$$m_j\frac{dv_r}{dt} = q_jE_r + \frac{l^2}{m_jr^3}, \quad l = m_jrv_\theta = constant \tag{4}$$

$$\frac{dr}{dt} = v_r \tag{5}$$

$$\rho = \sum_j q_j\delta(r - r_j) \tag{6}$$

In Eq. (4) l is the angular momentum of the particle and the additional term in the momentum equation is the centrifugal force.

3.2 Space and Time Differencing

We begin by first discussing the spatial differencing for these equations. The method we describe here is a simplified one dimensional form of that used in the new two dimensional code, CELESTE, developed by Brackbill and Forslund. We consider a set of N mesh points, $r(1), \ldots, r(N)$, that define the cell centers. These points can be arbitrarily spaced although the rate of change should be small compared to the cell size in order for the differencing to be reasonably accurate. We first consider how to deposit charge onto this grid, as shown in Eq. (6), in order to solve Poisson's equation, Eq. (3). The simplest procedure is to consider the particles to have zero size and to deposit all of the charge onto the grid point that is closest to the particle. This procedure is known as the nearest grid point (NGP) method. Although simple, it introduces large fluctuations on the scale of the grid point spacing as the particles move from grid point to grid point. These fluctuations are a form of collisions as mentioned in Section 2. Instead we take the particles to be of finite size and deposit charge on the grid in proportion to the overlap of the particle with each cell. Although a variety of particle sizes are possible, *Langdon and Birdsall* [*1970*] have shown that a particle size comparable with the cell size introduces the least amount of numerical error. For simplicity of calculation we choose a particle to be the size of the cell in which it is located. Since the grid spacing can vary with the index i, the particle size can change as it moves through the grid. However, the charge on a particle does not change. The shape of the particle is defined by the interpolation formula to and from the grid. For a uniform mesh a gaussian shape for the particle has also been used successfully[*Lin, et al., 1974*].

We use logical or natural coordinates, x_p, to describe the position of the particles such that the cell index is given by:

$$i = integer(x_p). \tag{8}$$

and the fractional part of x_p determines where in the cell the particle is located. The weights applied to cells i and $i+1$ are given by:

$$w_{i+1} = x_p - i,$$

$$w_i = 1.0 - w_{i+1} \tag{9}$$

The particle position then is converted to physical coordinates by:

$$r_p = w_i r_i + w_{i+1} r_{i+1}. \tag{10}$$

Keeping the logical position instead of the entirely equivalent physical one makes it easier to determine the cell location of the particle. The reverse process is more complicated and is done only once per time step. Similarly, the electric field, E_p, used to advance the particle velocity is interpolated with the same weight functions from the grid. The time centered acceleration and particle position advancement are given by:

$$v_p^{j+1/2} = v_p^{j-1/2} + \left(\frac{q_p}{m_p} E_p^j + \frac{l^2}{m_p^2 (r^j)^3} \right) dt \tag{11}$$

$$r_p^{j+1} = r_p^j + v_p^{j+1/2} dt \tag{12}$$

where the superscript j is the time step index. The quantity, v_θ, can be calculated from the conserved angular momentum at any time. We then use a rapid zero point search to determine the new natural coordinate of the particle. If the particle does not change cells, this can be done with only one iteration. With this new natural coordinate, new values of w_i and w_{i+1} are calculated and the charge is interpolated onto the grid:

$$q_i = q_i + w_i q_p$$

$$q_{i+1} = q_{i+1} + w_{i+1} q_p \tag{13}$$

If desired the grid can be moved before this new natural coordinate of the particle is calculated without introducing any diffusion into the system. Adding up the charge from all the particles yields the total charge at each grid point. The solution of Poisson's equation gives E^{j+1} and the time cycle is completed.

The solution of Poisson's Equation in one dimension is simple. In differenced form the equation is:

$$(r^2E)_{i+1/2} = (r^2E)_{i-1/2} + 4\pi(\rho r^2 dr)_i \tag{14}$$

The term $\rho r^2 dr$ is the total charge in a cell calculated above. The cell centered electric field then is obtained by averaging the quantity r^2E at the cell edges:

$$E_i = \left((r^2E)_{i+1/2} + (r^2E)_{i-1/2}\right) / \left(r^2_{i+1/2} + r^2_{i-1/2}\right) \tag{15}$$

where $r_{i+1/2} = (r_i + r_{i+1})/2$. Boundary conditions are applied by specifying E at the edge of the first cell. Alternatively Poisson's equation could be written in terms of the electrostatic potential and solved with a tridiagonal solver and center differenced to obtain E at the cell centers.

3.3 Boundary Conditions

An additional problem exists during the charge accumulation stage. Due to the finite size of the particles, some charge will be deposited outside the grid from cells at the edge of the system. An extra cell is added at each end to contain this charge. The charge accumulated in these "ghost" cells is added back onto the physical grid. In the Cartesian case one may also choose to have periodic boundary conditions, in which case the charge would be added into the first real cells at opposite ends of the grid.

Finally, the particles themselves must be checked for crossing of the system boundaries so that they may be absorbed, reflected, or recreated at some new velocity. These operations are done right after the new particle position is calculated before the new natural coordinates are calculated for redeposition of charge onto the grid.

This completes the basic description of a simple one dimensional electrostatic code. The algorithms described above are readily extendable to two dimensions. The appropriate cell size, time step, and number of particles that need to be used for any given problem is determined by numerical stability and accuracy considerations discussed below.

3.4 Particle Loading

We digress for a moment on how to load the particles with an arbitrary density distribution. Suppose we have N simulation particles which we would like to represent the initial arbitrary charge density, $\rho(r)$. The probability of a particle being at the position r is given by

$$P(r) = \frac{\int_{r_1}^{r} \rho(r')r'^2 dr'}{\int_{r_1}^{r_2} \rho(r')r'^2 dr'} \tag{16}$$

where r_2 and r_1 are the upper and lower limits of the system. We can then randomly pick a number P from 0 → 1 for each simulation particle and use the inverse function r(P) to give the particle position. There is less noise in the initial loading. however, if one uniformly divides the interval 0 → 1 into N parts. The charge associated with each of the particles is

$$q_p = \frac{1}{N} \int_{r_1}^{r_2} \rho r^2 dr \tag{17}$$

which, because of the small value of N, is much larger than the charge on an ion or electron. That is, we see that each particle here represents a collection of a large number of ions or electrons and hence is sometimes called a macroparticle. This loading technique is readily extendable to two dimensions for arbitrary two dimensional density distributions and may also be used for generating particles with some specified velocity distribution.

4. Data Management

Typically fo: large scale plasma simulations, modern computers do not have a enough larg· memory to contain all of the particle data. Since external storage is usually slow compared to the fast memory in a modern supercomputer, a well-tuned multi-channel input/output (I/O) pac!:age is needed to interface with this slow storage (usually disks). The fastest method is not always the best due to the environment of other users on the machine and the algorithms used to charge for computer resources. However, generally synchronous I/O (in which the I/O is going on simultaneous with the computing) on at least one channel each for input and output is required. The simplest form of I/O divides a buffer into three parts doing input into 1/3, computing on 1/3, and output on 1/3. In a multi- channel method there may be separate buffer for input and output. The package written for WAVE code on the CRAY-X/MP computers issues multiple simultaneous I/O requests on each channel and packs the particle data in order to maximize the I/O rate for a given buffer size. An interrupt driven I/O scheme has not been found to be fast enough to obtain data from the disk without missing revolutions on the disks. Also available on some modern supercomputers is an intermediate speed memory device (the SSD on the CRAY- X/MP) which is usually more than adequate to keep up with the particle pushing. In this case a sophisticated I/O algorithm is not required.

5. Optimization

5.1 Vectorization

Most modern supercomputers have very high speed vector performance compared with their scalar performance. It is important in the coding process to ensure that the critical sections of the code are as vectorized as possible. Within the particle mover itself most of the operations are vectorizable. For example, in the one dimensional electrostatic code described above Eqs. (7), (8), (10), and (11) are ·omplete vectorizable. However, because the cell index is random from one particle to the next, the interpolation from and to the grid (Eqs. (9) and (12)) is not. These two parts need to be in separate do loops from the vectorizable operations to ensure maximum speed. The electric field accumulation in Eq. (13) is also not vectorizable.

On the CRAY-X/MP a new hardware instruction has been added which now allows the interpolation of Eq. (9) to be completely vectorized. However, the scatter operation of Eqs. (12) remains non-vectorizable because of the accumulation. Supercomputers which have this hardware gather-scatter and a fast vector sum routine would allow complete vectorization of the particle mover.

5.2 Assembly Language

In the WAVE code the particle movers have been hand coded in assembly language to make optimal use of the CRAY-X/MP. We find that both the vector and scalar operations are about twice as fast as they are in FORTRAN. This should always be considered when the code is going to be used heavily with infrequent changes. If a careful isolation of parameters for the particle mover is done, the assembly language code will not have to be modified often even though the remainder of the code may change a great deal. With the WAVE code this modularity has allowed the same assembly language movers to be used for over 8 years on the CRAY-1 with essentially no changes, even though the remainder of the WAVE code has been almost entirely rewritten.

5.3 Multiprocessing

With the arrival of multiple processors in supercomputers it is possible to reduce the running time by processing more than one group of particles at a time. This does not reduce the CPU time of a given job (it actually increases it slightly) but may reduce the turn around time when this is most important. However, care must be taken because multiple groups of particles may try to access a shared piece of memory at the same time. In particular, the charge weighting process may require separate copies of the charge array for each task followed by a combining at the end of the cycle to avoid overwriting data. Some computer hardware does allow for a memory lock out when another processor is accessing the memory, but this still could result in some inefficiency. In the CELESTE code we have sought to overcome this problem by sorting the particles by means of linked lists. There is some additional overhead for this, but it allows for complete separation of particle pushing tasks. In addition, it allows for the inclusion of momentum and energy conserving binary collisions in a fully vectorized form.

6. Numerical Stability

6.1 Time Step Constraints

The discretization of time and space in simulation codes introduces accuracy and stability conditions. In the usual explicit particle code in which information at time level j is used to advance the fields and particles to $j+1$ there is a limit to the size of the time step before the code becomes unstable due to time aliasing. This is usually $\omega_p \delta t < 2$. Therefore, one cannot avoid the electron dynamics in an explicit plasma simulation code by taking very large time steps. One must have a time step large enough to resolve the highest frequency in the system.

For electromagnetic codes including the displacement current one has an additional constraint that can be even more severe. The timestep must satisfy the Courant condition, $\delta t \leq c\delta x$. Again this arises because of the necessity of resolving the highest frequencies in the system. Other instabilities involving Cerenkov processes have also been shown to exist[*Godfrey, 1974,1975*]

6.2 Finite Grid Instability

The aliases arising from the spatial grid are coupled together by the interpolation method in a complicated way. In general, it is found that if the Debye length, $\lambda_d \geq \delta x/2$, the numerical instabilities are almost entirely absent both for drifting and stationary plasmas because modes with the largest wavenumber, $k_{max}\delta x \sim 1$ are heavily Landau damped. For large δx the incorrect of damping on the aliases can overcome the physical Landau damping at k_{max} and hence can cause strong instability. The addition of strong smoothing at large k can alleviate these effects somewhat.

The properties of the numerical instabilities depend on the details of the particular differencing scheme. *H. R. Lewis*[*1970*] has suggested a differencing scheme derived from the system Hamiltonian that conserves energy in the limit of $\delta t \rightarrow 0$. The only difference from the method described above that conserves momentum exactly but not energy is that E is defined at the cell edge and the particle velocities are advanced with the NGP method. As one might expect, the method is somewhat noisier than the usual method and, in practice, does not conserve energy well because of the finite δt. If, however, one introduces smoothing on the charge density, the two methods conserve energy about equally well. For the physically stable case of a Maxwellian electron plasma drifting through a fixed ion background on a uniformly space Cartesian grid with periodic boundary conditions, we compare the stability properties [*Lindman, 1970; Langdon, 1970a,1970b; Okuda, 1972*] of the two methods. The energy conserving method is stable for zero drift while the momentum conserving method is not. For drift velocities about equal to the electron thermal speed, the energy conserving method becomes unstable and the growth rate of the momentum conserving method diminishes sharply. For $\delta x = 10\lambda_d$, the maximum growth rates in both cases are about $0.02\omega_p$at $k\lambda_d = .16$. For smaller grid size the growth rate drops off sharply.

7. Two Dimensional Electromagnetic Codes

7.1 Units

We now discuss the basic difference equations for electromagnetic plasma simulation in two dimensions. Before considering the difference equations, let us first discuss the system of units used in the two dimensional WAVE code[*Morse and Nielson, 1971*]. We find it useful to normalize the density to some charge density, n_o, the charge to mass ratio to some q_o/m_o and velocities to the speed of light. Time then is normalized to $\omega_p = (4\pi n_o q_o^2/m_o)^{1/2}$, the electric and magnetic fields are in units of $(4\pi n_o m_o c^2)^{1/2}$ and the potentials in units of $m_o c^2/q_o$. Maxwell's equations then have no numerical factors and are dimensionless. Several additional nontrivial advantages result from this system of units. The equations are automatically ordered so that one can compare the size of various terms. Also, the dimensionless equations imply that a given simulation is the solution to a whole class of physical problems that are scaled by the units of time and fields. This reduces the likelihood of unnecessarily running a whole series of equivalent problems. It is easy then to convert the scaled units into the physical units of a problem of interest. Doing this at the outset can shed considerable insight into the physics.

7.2 Field Equations

The interpolation techniques used in one dimension are immediately extended to two dimensions. For simplicity we consider the case of a uniformly spaced mesh in which the particle is the same size as the cell spacing. Here the amount of charge and current deposited into each cell is proportional to the area overlap in the cell, hence the term "area weighting" or bilinear interpolation.

In the WAVE code the equations are written in term of the vector and scalar potentials for historical reasons. *Langdon and Lasinski* [*1976*] have shown that an E and B based code has essentially identical properties provided the equations are differenced properly. Although an explicit wave equation solution exists in the WAVE code, we describe here the difference equations that allow for δt to exceed the Courant limit by controlling the high frequencies allowed in the system[*Nielson and Lindman, 1972*]. The equations for Ampere's Law and Faraday's Law give rise to the wave equation for $\vec{A}$:

$$\begin{aligned}&\nabla^2(\vec{A}^{3/2} + 2\vec{A}^{1/2} + \vec{A}^{-1/2})/4 \\ &- (1 + \beta D^2\nabla^2)(\vec{A}^{3/2} - 2\vec{A}^{1/2} + \vec{A}^{-1/2})/c^2(\delta t)^2 = -\vec{J}_T^{1/2}\end{aligned} \tag{18}$$

where

$$\nabla^2\vec{A} = \frac{\vec{A}_{j+1,m} - 2\vec{A}_{j,m} + \vec{A}_{j-1,m}}{(\delta x)^2} + \frac{\vec{A}_{j,m+1} - 2\vec{A}_{j,m} + \vec{A}_{j,m-1}}{(\delta y)^2} \tag{19}$$

and

$$\frac{1}{D^2} = \frac{1}{(\delta x)^2} + \frac{1}{(\delta y)^2} \tag{20}$$

and

$$0 \le \beta \le 0.25(\delta t)^2/D^2. \tag{21}$$

The superscripts indicate the time levels of the various terms. These equations, of course, are solved for the 3/2 time level using the earlier values as input. The parameter β is used to alter the dispersion relation so that timesteps larger than the Courant limit can be used. The Coulomb gauge is used so that the transverse current $\vec{J}_T$, is given by:

$$\vec{J}_T = \vec{J} - \nabla\frac{\partial\phi}{\partial t}. \tag{22}$$

In order to guarantee that $\nabla \cdot \vec{A} = 0$ and $\nabla \cdot \vec{J}_T = 0$ so that charge is conserved at each time step, we must solve the auxiliary equation:

$$\nabla^2\frac{\partial\phi}{\partial t} = \nabla \cdot \vec{J}. \tag{23}$$

where ϕ is at the whole time step and $\vec{J}$ is at the half time step and use thise $\partial\phi/\partial t$ in Eq. (22). The fields $\vec{E}$ and $\vec{B}$ are computed by:

$$\begin{aligned} \vec{E}^1 &= -\nabla\phi^1 - (\vec{A}^{3/2} - \vec{A}^{1/2})/\delta t \\ \vec{B}^1 &= \nabla \times (\vec{A}^{3/2} + \vec{A}^{1/2})/2 \end{aligned} \tag{24}$$

The electrostatic potential is obtained from

$$\nabla^2\phi^1 = \rho^1 \tag{25}$$

All fields, charge and currents are defined at cell centers. Given the currents at the half time step and the charge at the whole time step the fields can be advanced to the next whole time step with the vector potential being one half time step ahead.

The field equations are solved with a direct two dimensional Poisson solve. The method proceeds by Fast Fourier Transform in the y direction followed by Gaussian elimination in the x direction on the linearly independent Fourier components. The result is then Fourier transformed back to physical space. This allows for arbitrary boundary conditions in x. General boundary conditions in y are obtained by choosing an appropriate set of basis functions for the FFT algorithm. Boundary conditions that have been implemented in the y direction besides periodic include Neumann and Dirichlet for $\vec{A}$ and ϕ. Even more general boundary conditions are available in x.

7.3 Particle Equations

The particle velocities are known at the half time step while $\vec{r}$, $\vec{E}$, and$\vec{B}$ are known at the whole time step. If we define $h = q\delta t/m$, the centered form for the equation of motion [*Morse and Nielson, 1971*] is:

$$\vec{v}^{3/2} = \vec{v}^{1/2} + h(\vec{E}^1 + \vec{v}^1 \times \vec{B}^1) \tag{26}$$

However, we do not yet know $\vec{v}^1$. By using $\vec{v}^1 = \frac{1}{2}(\vec{v}^{3/2} + \vec{v}^{1/2})$, we can solve this equation for $\vec{v}^{3/2}$ implicitly. We do this by taking the dot product and cross product with $\vec{B}^1$. The time centered reversible equation then is

$$\begin{aligned} \vec{v}^{3/2} = \frac{1}{1 + \frac{h^2}{4}\vec{B}\cdot\vec{B}} \Bigg(& \vec{v}^{1/2}\left(1 - \frac{h^2}{4}\vec{B}\cdot\vec{B}\right) + h(\vec{E} + \vec{v}^{1/2} \times \vec{B}) + \frac{h^2}{2}\vec{E} \times \vec{B} \\ & + \frac{h^2}{2}\vec{B}\vec{v}^{1/2}\cdot\vec{B} + \frac{h^3}{4}\vec{B}(\vec{E}\cdot\vec{B})\Bigg) \end{aligned} \tag{27}$$

If we wish to keep terms only to order $(\delta t)^2$ we can drop the last term. For the relativistic equations a similar expression can be derived. The position advancement is as it was in one dimension. The velocities are interpolated onto the grid at a position corresponding to the advance of one- half time step to obtain $J^{1/2}$. The charges are interpolated on the grid at the whole time step.

8. Implicit Techniques

8.1 Motivation

In studying the nonlinear properties of many plasma systems we are primarily interested in the long time and long scale length plasma behavior. For example, in collisionless shocks we frequently are interested in time scales long compared to the ion gyro frequency and long compared to the ion gyro radius. For space plasmas there are over 10^3 Debye lengths in an ion gyro radius and over 10^4 plasma periods in an ion gyro period. To resolve all of these time and space scales is prohibitive even with the next generation supercomputers. There are some problems in which the high frequencies and short scale lengths communicate effectively with the long wavelength fluctuations. If this is the case, one cannot make any significant simplifications. However, frequently the highest frequencies and shortest scalelengths are not important for the evolution of the longest time and space scales. Here several solutions are possible.

A common techniques is to solve a reduced set of equations for one of the components of the plasma system. This is usually referred to as a hybrid method. There are a variety of hybrid models that have had considerable success in modeling a wide variety of plasma phenomena. In recent years, however, implicit techniques have undergone considerable development [*Brackbill and Forslund, 1982; Friedman, et al., 1982; Langdon, et al., 1983*]. By implicit we mean that the equations are solved backward in time from time level $j+1$ to time level j. This results in a decaying of the solutions that would be unstably growing in an explicit formulation. A simple example of this was given above for the wave equation solved in the two dimensional electromagnetic code WAVE. Although this technique is well known in hydrodynamics and magnetohydrodynamics, it has only recently been found to be practical in plasma simulation. The difficulty arises because there are many more simultaneous equations to be solved in the plasma system. If one were to have to iterate on all of these equations together, the time would be prohibitive and there would be no advantage over simpler explicit methods.

8.2 Outline of Methods

There is not enough space here to discuss the details of the implicit methods which have been developed. Other papers in this volume will deal with this in more detail as will a forthcoming book in the Computational Techniques series by Academic Press entitled 'Timescales" to be published in 1985. With some oversimplification it can be stated that it has been found that one does not have to iterate the simultaneous set of equations together to obtain a stable accurate system. There are several approaches to this problem, but with a modified set of equations for the particles and fields one can obtain a stable system. For example, in the moment method [*Mason, 1981; Denavit, 1981*] a fluid-like description of the plasma is solved for implicitly with conventional implicit techniques. The particles then are advanced with these new fields and used to correct the moments which were obtained from the fluid-like treatment. At the same time the implicit solution of the fluid equations over a large time step removes the high frequency noise injected by the particle moments. Thus the two stages serve to correct each other. It has been shown [*Brackbill and Forslund, 1982*] that this results in a systematic elimination of the short wavelength and high frequency waves in the system allowing for large time and space steps. This method has been particularly successful in studying the self-consistent turbulence generated in magnetic shock waves[*Quest, et al., 1983; Forslund, et al., 1984*]. The direct method [*Friedman, et al., 1981, Langdon, et al., 1983*] which is derived on more formal grounds has similar properties to the moment method although all the differences have yet to be discussed in the literature. In some of the formulations of these implicit methods there are still some time and space step constraints which limit the application of the method, although it is not clear if these are in principle limitations or simply implementation limitations.

9. Problems solved

The number of problems which have been solved by means of particle simulation are too numerous to cite accurately in this limited space. However the range has extended from space physics to laser fusion to magnetic confinement fusion, from collisionless shock waves to parametric instabilities to electron transport. For example, an important mechanism causing an anomalous resistivity in magnetic shock waves has been identified with two dimensional implicit calculations with the implicit code, VENUS [*Forslund, et al., 1984*]. Magnetic field induced electron transport on the surface of laser fusion targets was first discovered with plasma simulation[*Forslund and Brackbill, 1982*]. Most of the information known about how energetic electrons are produced by intense laser light have been obtained by plasma simulation[*Forslund, et al., 1977*]. Recently in the area of particle acceleration by a beating between two light waves of different frequencies, limitations due to self-focusing, filamentation, back scattering and magnetic field generation have all been identified in simulation before they could be observed experimentally[*Joshi, et al., 1984*]. Frequently the results from simulations were unexpected but are easily understood in simple terms once they are known. This has been a typical path of research in plasma simulation, a field which has only begun to develop.

References

Brackbill, J. U. and D. W. Forslund, *J. Comput. Phys.*, **46**, 271, 1982.

Brackbill, J. U., D. W. Forslund, K. B. Quest, and D. Winske, *Phys. Fluids*, **27**, 2682, 1984.

Buneman, O., *J. Comput. Phys.*, **1**, 517, 1967.

Denavit, J., *J. Comput. Phys.*, **9**, 75, 1972.

Denavit, J., *J. Comput. Phys.*, **47**, 337, 1981.

Forslund, D. W. and J. U. Brackbill, *Phys. Rev. Lett.*, **48**, 1614, 1982.

Forslund, D. W., J. M. Kindel, and K. Lee, *Phys. Rev. Lett.*, **39**, 284, 1977.

Forslund, D. W., K. B. Quest, J. U. Brackbill, and K. Lee, *J. Geophys. Res.*, **89**, 2142, 1984.

Friedman, A., A. B. Langdon, and B.I. Cohen, *Comm. Plasma Phys. Contr. Fusion*, **6**, 225, 1981.

Godfrey, B. B., *J. Comput. Phys.*, **15**, 504, 1974.

Godfrey, B. B., *J. Comput Phys.*, **19**, 58, 1975.

Joshi, C., W. B. Mori, T. Katsouleas, J. M. Dawson, J. M. Kindel, and D. W. Forslund, *Nature*, **311**, 525, 1984.

Langdon, A. B., *J. Comput. Phys.*, **6**, 247, 1970a.

Langdon, A. B., *Proc. 4th Conf. on Num. Sim. of Plasmas*, 1970b.

Langdon, A. B., *J. Comput. Phys.*, **12**, 247, 1973.

Langdon, A. B., and C. K.Birdsall, *Phys. Fluids*, **13**, 2115, 1970.

Langdon, A. B., and B. F. Lasinski, *Methods Comput. Phys.*, **16**, 327, 1976.

Langdon, A. B., B. I. Cohen, and A. Friedman, *J. Comput. Phys.*, **51**, 107, 1983.

Lewis, H. R., *J. Comput. Phys.*, **6**, 126, 1970.

Lin, A. T., J. M. Dawson, and H. Okuda, *Phys. Fluids*, **17**, 1995, 1974.

Lindman, E. L., *J. Comput. Phys.*, **5**, 13, 1970.

Mason, R. J., *J. Comput. Phys.*, **47**, 233, 1981.

Morse, R. L. and C. W. Nielson, *Phys. Fluids*, **14**, 830, 1971.

Nielson, C. W. and E. L. Lindman, *Proc. 6th Conf. on Num. Sim. of Plasmas*, 1972.

Okuda, H., *J. Comput. Phys.*, **10**, 475, 1972.

Quest, K. B., D. W. Forslund, J.U.Brackbill, and K. Lee, *Geophys. Res. Lett.*, **10**, 471, 1983.

Sakanaka. P. H., et al., *Phys. Fluids*, **14**, 611, 1971.